REMARKS

Claims 1-20 were presented for examination and all have been rejected under 35 U.S.C. §§ 112, 102, or 103. Claims 1, 4-10, 13-18, and 19 are being amended. Claims 3 and 12 are canceled, and claims 21-28 are being added. In view of the above amendments and following remarks, reconsideration of the application is respectfully requested.

CLAIM OBJECTIONS

In paragraph 2 of the Office Action, claim 6 and 15 were objected to because of the word "with" was missing in the phrase "a configuration file associated the first program." In response, claims 6 and 15 are being amended to recite "a configuration file associated with the first program." Withdraw of the objection is therefore respectfully solicited.

In paragraph 3, claim 16 was objected to because informalities in the phrase "wherein the commands are input . . . ." Claim 16 is being amended to claim in the alternative language, which also recites "wherein the debugging command is inputted . . . ." as suggested in the Office Action. Withdrawal of the objection is therefore solicited.

REJECTIONS UNDER 35 U.S.C. § 112

In paragraph 4 and 5, claims 9 and 18 were rejected as being indefinite because an antecedent basis was missing for the term "the Integrated Development Environment." In response, claims 9 and 18 are being amended to recite "using an Integrated Development Environment." Withdrawal of the rejection is therefore respectfully solicited.

REJECTIONS UNDER 35 U.S.C. § 102(e) – Gorshkov

In paragraphs 6 and 7, claims 1-6, 8-15, and 17-20 were rejected under 35

U.S.C. § 102 (e) as being anticipated by U.S. patent number 6,490,721 to Gorshkov

("Gorshkov").

In response to the claimed element "a method for augmenting a debugger

having debugging functionality used to debug a first program" the Office Action cited

Gorshkov's paragraphs of column 2, lines 58-64 and column 3, lines 29-53, which

disclose a debugging subprogram and a target program to be debugged. The Office

Action thus equated the claimed debugger and first program to Gorshkov's debugging

subprogram and target program to be debugged, respectively.

In response to the claimed element "providing a second program having

second-program functionality," the Office Action asserted "Figure 1 illustrates the

primary components of the preferred embodiment. Target program 10 is the existing

program in executable format that is to be debugged. . . . User action libraries 12 are

created by compiling one or more debugging subprograms 16 (containing user

actions) into linkable libraries (similar to Dynamic Link Library (DLL) in a

Window™ environment). This compiling procedure is accomplished by action

compiler 14 which compiles the source code of subprograms 16, which are written in

ANSI C for example, into machine code of user action libraries 12. These files

together with target program 10 are input to dynamic action linker 18. . . . Dynamic

action linker 18 reads target program 10 and user action libraries 12 and creates two

processes. A first process 20 created by dynamic action linker 18 consists of target

program 10 and the debugging user actions needed from user action libraries 12. A

second process 22 created by dynamic action linker 18 handles requests from process

20, to modify code locations in process 20 as described in detail below." in column 3, lines 29-53)."

From this lengthy citation, the Office Action failed to clearly correspond an element of Gorshkov to the claimed second program. In addition to other elements, this citation discloses both the first process 20 and the second process 22. However, regardless of whether the claimed second program was equated to the first process 20, the second process 22 or other elements in the cited paragraph, amended claim 1 recites that the piece of code to perform a task in response to a debugging command is selected from the second-program functionality if the breakpoint is an instrumentation breakpoint, which is not disclosed, suggested, or made obvious by Gorshkov. Even though Gorshkov's second process 22 "handles requests from process 20," Gorshkov does teach, suggest, or make obvious that the piece of code to perform a task is selected from the second-program functionality if the breakpoint is an instrumentation breakpoint.

In response to the claim element "providing integration code for analyzing commands used to debug the first program, and invoking appropriate pieces of code to perform tasks in responding to such commands; wherein the appropriate pieces of code are selected from one or a combination of functionality provided in a library, the debugging functionality, and the second-program functionality," the Office Action cited "[i]n step E, child process 22 has received the acknowledgment from parent process 20 of a successful attachment and continues on to patch into parent process 40 a call to the dynamic user actions runtime start routines, i.e. the debugging actions. In particular, calls to user actions in user action libraries 12 are inserted in the memory image of target program 10 (which is running as will be seen below) and user action libraries 12 are loaded into RAM in a separate area. In step F, child process waits for a new request for patching service from parent process 20. In step G, child process 22

has received a request from parent process 20 and executed the request. Steps F and

G are repeated until the task is terminated by parent process 20 . . . FIG. 3 illustrates

Step E of FIG. 2, i.e., patching of dynamic user actions into target program 10 in

detail. Child process 22 created by dynamic action linker 18 must patch the memory

image of target program 10 so that it will call the newly loaded user action routines.

In step E1, child process 22 allocates space for the patch in the patch area in parent

process 20. In step E2, child process 22 replaces an instruction (or instructions) at the

requested program location with a branch instruction to the patch area. In step E3,

child process 22 generates code to call the user action." in column 3, line 67 to

column 4, lines 1-13; and column 4, lines 29-38."

Again, the Office Action combined many of the claimed elements into one

lengthy response and thus failed to correspond each element of the claim to an

element of Gorshkov. The claimed elements include, for example, integration code,

appropriate pieces of code to perform tasks, the selection of the pieces of code, etc.

However, claim 1 is being amended to claim in the alternative language, and

include elements clearly not disclosed, suggested, or made obvious by Gorshkov. For

example, Gorshkov does not disclose, suggest, or make obvious the integration code,

which invokes a piece of code to perform a task in response to a debugging command,

based on types of a breakpoint. Additionally, Gorshkov does not disclose, suggest or

make obvious "wherein if the breakpoint is a debugging breakpoint, then the piece of

code is selected from the debugging functionality, else if the breakpoint is an

instrumentation breakpoint, then the piece of code is selected from the second-

program functionality."

Because claim 1 recites limitations patentably distinguished from Gorshkov,

claim 1 is patentable.

Claims 2-9 depend directly or indirectly from claim 1 and are therefore patentable for at least the same reasons as claim 1.

Claims 2-9 are also patentable for their additional limitations.

Regarding claim 2, the Office Action asserted that Gorshkov's paragraph of column 4, lines 29-38 discloses "the step of using an instrumentor as the second program." However, this cited paragraph discloses patching dynamic user actions into target program 10; child process 22 must patch the memory image of target program 10 so that it will call the newly loaded action routines; child process 22 allocates space for the patch in the patch area in parent process 20; child process 22 replaces an instruction, generates code to the user action, etc. As can be seen, this cited paragraph does not disclose an instrumentor being used as the second program, and patching dynamic user actions into a target program is not patentably comparable to using an instrumentor as the second program. Further, as claimed, the second program or the instrumentor includes its functionality, and if the breakpoint is an instrumentation breakpoint, then the piece of code to perform a task in response to a debugging command selected from the functionality of the instrumentor.

Regarding claim 3, the cited paragraph of column 4, lines 29-38 does not disclose the claimed "using a first piece of code having the appropriate pieces of code to perform the task." However, claim 3 is being canceled in conjunction with amendments in other claims.

Regarding claim 4, the Office Action asserted that Gorshkov's paragraph of column 4, lines 29-38 discloses the method "further comprises the step of making the first piece of code an executable part of the first program." The cited paragraph does not disclose this claimed element. However, claim 4 is being amended to depend on claim 1 instead of claim 3 and recites "[t]he method of claim 1 further comprises the step of making the piece of code an executable part of the first program." Even

though Gorshkov's cited paragraph discloses "patching of dynamic user actions into target program 10," because the dynamic user actions of Gorshkov do not correspond to the claimed "piece of code to perform a task in response to a debugging command, based on types of a breakpoint," making the piece of code an executable part of the first program is patentably distinguished from patching dynamic user actions into target program 10.

Regarding claim 5, the Office Action asserted that Gorshkov's paragraph of column 4, lines 29-38 discloses "further comprises the step of using a trampoline as the first piece of code." The cited paragraph does not disclose this claimed element. However, claim 5 is being amended to depend on claim 1 instead of claim 3 and recites "[t]he method of claim 1 further comprises the step of using a trampoline as the piece of code." The cited paragraph discusses patching of user actions into target program 10, but does not disclose a trampoline, and therefore cannot disclose using a trampoline as the piece of code that performs a task in response to a debugging command based on types of a breakpoint.

Regarding claim 6, the Office Action asserted that Gorshkov's paragraph of column 3, lines 29-53 discloses that "wherein the commands are selected from one or a combination of: input from a user using the debugger; a script file associated with the first program; and a configuration file associated with the first program." Gorshkov's cited paragraph discusses that target program 10 was compiled from source code; user action libraries are created by compiling one or more debugging subprograms, but has nothing to do with a debugging command being selected from one or a combination of input from a user using the debugger, a script file associated with the first program, a configuration file associated with the first program. If the target program 10 corresponds to the first program, then, to be parallel with claim 6, Gorshkov must disclose the debugging command is selected from a script file

associated with the target program, a configuration file associated with the target program, etc., but fails to do so.

Regarding claim 8, the Office Action asserted that Gorshkov's paragraph of column 3, lines 29-44 discloses integrating the debugger, the instrumentor, and the integration code into a combined code; and embedding the combined code into a language environment. This cited paragraph does not disclose that claimed element. However, claim 8 is being amended to claim in the alternative language of the instrumentor as the second program. Further, the cited paragraph discusses patching dynamic user actions into target program 10 and the function of the code patched into the parent process. The target program was equated to the first program. Combining the debugger, the instrumentor, and the integration code into a combined code is patentably distinguished from patching dynamic user actions into target program 10. Further, this cited paragraph does not disclose a language environment or embedding the combined code into the language environment. If the claimed combined code corresponds to the target program 10, then, to be parallel with claim 8, Gorshkov must disclose embedding the target program 10 into a language environment, but fails to do so.

Regarding claim 9, the Office Action asserted that Gorshkov's paragraph of column 3, lines 29-44 discloses using an Integrated Development Environment as the language environment. However, this cited paragraphs discusses that the target program 10 was compiled from source code, and user actions are created by compiling debugging subprograms into linkable libraries, but has nothing to do with the Integrated Development Environment, which, in an embodiment, includes Visual Studio, etc. (Applicant's Specification, page 13, line 2).

Claims 10-18 recites limitations corresponding to claims 1-9, and are therefore patentable for at least the same reasons as claims 1-9.

Claims 19 and 20 recite limitations corresponding to claims 1 and 2, and are therefore patentable for at least the same reasons as claims 1 and 2.

REJECTIONS UNDER 35 U.S.C. § 103(a) – Gorshkov in view of Kim

In paragraph 8 and 9, claims 7 and 16 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Gorshkov in view of U.S. patent number 6,003,143 to Kim ("Kim").

Kim does not provide the claimed elements missed from Korshkov, and claim 1 and claim 7 are therefore patentably distinguished from Korshkov and Kim, either alone or in combination. Further, the alleged motivation for combining Gorshkov and Kim is improper, and showing a prima facie case of obviousness failed. The assertion that "the modification [of Gorshkov to include the step of inputting the commands at a debugging prompt using the teaching of Kim] would be obvious because one of ordinary skill in the art would be motivated to allow users to *interactively* debug as the program is running" (emphasis added) is conclusory and a general statement related to Kim, but there is no suggestion in Gorshkov or Kim to combine the teaching of the two. In fact, Gorshkov teaches away from using the debugging prompt or interactive debugging of Kim because, according to Gorshkov it is the object of the invention "to permit debugging of a computer program without *stopping execution of the program*" (col. 2, lines 21-23, emphasis added) or "to permit *noninteractive* debugging of a computer program" (col. 2, lines 24-25, emphasis added).

<u>NEW CLAIMS</u>

Added claims 21-24 depend from claim 1 and are therefore patentable for at least the same reasons as claim 1. Claims 21-24 are also patentable for their own limitations.

Added claims 25-28 depend from claim 10 and are therefore patentable for at least the same reasons as claim 10. Claims 25-28 are also patentable for their own limitations.

Further, all limitations in the added claims are supported in the Specification and therefore no new matter is being added.
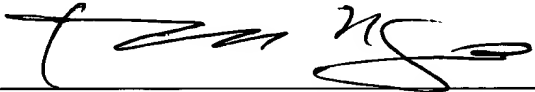
SUMMARY

In conclusion, it is respectfully submitted that pending claims 1-20 and added claims 21-28 clearly present subject matter that is patentable over the prior art of record, and therefore withdrawal of the rejections and consideration of the added claims are respectfully requested.

Respectfully submitted,

Date: _Dec 16, 2004_   By: _____

Tuan V. Ngo, Reg. No. 44,259
IP Administration
Legal Department, M/S 35
Hewlett-Packard Company
P. O. Box 272400
Fort Collins, CO 80527-2400
Phone (408) 447-8133
Fax (408) 447-0854